

[UW Home](#) > [Computing and Networking](#) > [Unix](#)

Quick Reference: vi

[Modes of Operation](#)

[Entering and Leaving vi Files](#)

[Basic Cursor Movement](#)

[Inserting Text](#)

[Deleting, Retrieving, and Undoing](#)

[Changing, Replacing, and Copying Text](#)

[Moving Around in a File](#)

[File Manipulation](#)

[Searching For Text](#)

[Setting Options](#)

[Common Option Settings](#)

Modes of Operation

vi, the Unix visual editor, has two modes of operation:

1. **Command mode**--This is vi's initial and normal state. In this mode, most commands you type are not displayed. Only commands preceded by **:**, **/**, or **!** are displayed on the status line, which also gives feedback about input, output, and large changes. Execute these commands with **<Return>** or **<Esc>**. Use **<Esc>** to change from text input mode to command mode (when in doubt, press **<Esc>**).
2. **Text input (Insert) mode**--This mode is entered by pressing **a**, **A**, **i**, **I**, **o**, or **O**. Press **<Esc>** to end text input mode and return to command mode.

For more information about using vi, see the online help note (on Uniform Access Unix computers, enter **help vi**).

The following notations are used in this document (variables appear in *italics*):

<i>x</i>	represents a single character
<Control> <i>x</i>	control character: while holding down the <Control> key, press <i>x</i>
<i>text</i>	one or more characters
<i>n</i>	represents a number
<i>pat</i>	text and pattern matching characters
<Return>	Return key on your terminal

<Esc>

Escape key on your terminal

Entering and Leaving vi Files

vi *filename(s)*

edit a file or files

vi -r *filename*

retrieve saved version of file after system or editor crash

vi -x *filename*

edit encrypted file

vi -wn *filename*set default window size to *n* (useful for dial-ups)**:wq**

save (write) file and exit to system prompt

ZZ

save file and exit to system prompt

:q!

discard all changes and exit to system prompt

Basic Cursor Movement

Use **h**, **j**, **k**, and **l** to move the cursor--using arrow keys may result in undesirable consequences**h**

move cursor left one character

j

move cursor down one line

k

move cursor up one line

l

move cursor right one character

(Any of the above commands preceded by *n* will move the cursor *n* spaces or lines in the indicated direction.)

Inserting Text

(If *n* precedes an insert character, *n* copies of inserted text are added upon escape from insertion mode.)**a**

begin insert at right of cursor

A

begin insert at end of line

i

begin insert at left of cursor

I

insert at beginning of line

o

open line below, ready for insertion

O

open line above, ready for insertion

S

replace text with blank line; begin insertion at beginning of that line

<Control>i

insert tab

<Control>v

insert non-printing character

<Backspace>

erase character (invisible until over-typed or insert mode escaped)

<Esc>

terminate insert mode; also terminates unwanted commands

Deleting, Retrieving, and Undoing

dw

delete word

dd

delete line

yw

yank word into buffer

yy

yank line into buffer

x

delete character

D

delete characters from cursor to end of line

ndwdelete *n* words into buffer***ndd***delete *n* lines into buffer***nyw***yank *n* words into buffer***nyy***yank *n* lines into buffer***nx***delete *n* characters into buffer**p**

put buffer contents after cursor

P

put buffer contents before cursor

u

undo last single change

U

restore current line

Changing, Replacing, and Copying Text

.

repeat last change

n.repeat last change *n* times***cwtext***mark end of a word with **\$** and change to *text* (press **<Esc>** to end)***rx***replace character under cursor with character *x****nrx***replace *n* characters with character *x****Rtext***write over existing *text*, (**<Esc>** to end)**J**

join succeeding line to current cursor line

:s/pat1/pat2

on the current line, substitute the first occurrence of pattern 1 with pattern 2

:s/pat1/pat2/g	on the current line, substitute all occurrences of pattern 1 with pattern 2
;&	;repeat the last :s request
:%s/pat1/pat2/g	substitute all occurrences of pattern 1 with pattern 2 throughout the file
::,\$s/pat1/pat2/g	substitute all occurrences of pattern 1 with pattern 2 from cursor to end of file

Moving Around in a File

<Control>g	ascertain line number of current line
G	go to end of file
nG	go to line <i>n</i>
<Return> or +	move cursor to beginning of next line
-	move to beginning of previous line
w or nw	move one word or <i>n</i> words to the right
b or nb	move one word or <i>n</i> words to the left
)	move to next sentence
(move to previous sentence
}	move to next paragraph
{	move to previous paragraph
<Control>d	scroll down one-half screen
<Control>u	scroll up one-half screen
<Control>l	clear and redraw the screen
m <i>x</i>	mark cursor position with character <i>x</i>
` <i>x</i>	move to position marked with <i>x</i>
d` <i>x</i>	delete text from marked <i>x</i> to cursor
y` <i>x</i>	yank text from marked <i>x</i> to cursor

Note: If you precede the mark letter with ' (apostrophe) instead of ` (grave accent), the action will apply to the entire line the mark is in, not the exact marked location.

File Manipulation

:rfile	read in a file beginning on the line below the current line
:w	save and remain in current file

:wq	save file and quit
:q	quit (leave unedited file)
:q!	quit and do not save changes
!:command	run single Unix command while editing (press <Return> to return to file)
:sh	obtain temporary shell (<Control>D to return to file being edited)
:n,mml	move lines numbered <i>n</i> through <i>m</i> after line <i>l</i>
:n,m\lll	make a copy of lines numbered <i>n</i> through <i>m</i> and put after line <i>l</i>
:n,mwfile	write lines numbered <i>n</i> through <i>m</i> to <i>file</i>
:n,mw>>file	append lines numbered <i>n</i> through <i>m</i> to end of <i>file</i>
:'a,'bwfile	write block, marked with <i>a</i> and <i>b</i> , to <i>file</i>

Searching For Text

/pat	go to pattern <i>pat</i> (forward in file from current cursor position)
?pat	go to pattern <i>pat</i> (backward in file from current cursor position)
n	repeat last search, looking in direction of initial search
N	repeat last search, looking in reverse direction of initial search
%	find matching () or { } or [] (can be used in combination with / , ? , n , or N to search for matching brackets throughout file)

Setting Options

Options are either toggled on and off, or given values. When editing, set options for a file with the **set** command. If you want options to be permanent in a particular directory, create a **.exrc** file in that directory and set options in that file: **set option option option=n**. (For example, **set ai sm sw=4**.) If you want certain option settings to apply throughout your Unix environment, edit your **.login** file by entering **setenv EXINIT'set option option option=n'** (for example, **setenv EXINIT'set ai sw wm=10'**).

The values in a **.exrc** file for a directory will override the values of EXINIT in the **.login** file. If no **.exrc** file exists, any option values set in the **.login** are used. If some option values are set in the **.exrc** file and others are set in the **.login** file, values from both files are used.

:set all	displays all option settings on your terminal
:set	displays settings set by EXINIT , the .exrc file, and any current changes

:set option	sets option
:set option=n	sets option and assigns it the value of <i>n</i>
:set nooption	unset option
:set option?	displays setting of option on screen status line

Common Option Selections

(To see a complete list of options, enter **:set all**.)

<u>Option Name</u>	<u>Default</u>	<u>What Option Does</u>
autoindent (ai)	noai	provides automatic indentation during text entry
autowrite (aw)	noaw	automatically saves file (write) before searches, control codes, escapes to shell
ignorecase (ic)	noic	ignore case during searches
lisp	nolisp	modify)({ } [] to be compatible with <i>lisp</i>
list	nolist	show tabs (^I) and ends of lines (\$)
magic	magic	allows metacharacters; with nomagic , these only include <Control> and \$
number (nu)	nonu	show line numbers
readonly (ro)	norro	make file status read only
redraw (re)	nore	simulate smart terminal on dumb
shell	sh=/bin/sh	pathname of new shell for ! and :sh (default from \$SHELL if present)
showmatch (sm)	nosm	show matching (or { when) or } is entered
term	\$TERM	name of terminal being used; set by Unix \$TERM
terse	noterse	provide shorter error diagnostics
wrapmargin (wm)	wm=0	cause lines to be broken at least <i>n</i> spaces from right edge of screen



[Computing & Communications](http://cac.washington.edu) help@cac.washington.edu Modified: January 23, 2002